2022 OPJU International Technology Conference on Emerging Technologies for Sustainable Development (OTCON) | 978-1-6654-9294-2/23/$31.00 ©2023 IEEE | DOI: 10.1109/OTCON56053.2023.10113988

Success and failure rate prediction of Android Application using Machine Learning

Swagatam Jay Sankar
School of Electronic Engineering,
KIIT Deemed to be University,
Bhubaneswar 751024, Odisha, India
swagatamjaysankar02@gmail.com

Utkrisht Singh
School of Computer Engineering,
KIIT Deemed to be University,
Bhubaneswar 751024, Odisha, India
utkrisht0001@gmail.com

Israj Ali
School of electronics Engineering,
KIIT Deemed to be University,
Bhubaneswar 751024, Odisha, India
israjfet@kiit.ac.in

M.Nazma Naskar
School of Computer Engineering,KIIT Deemed to be University, Bhubaneswar 751024, Odisha, India
naskar.preeti@gmail.com

Mahendra Kumar Gourisaria
School of Computer Engineering,
KIIT Deemed to be University,
Bhubaneswar 751024, Odisha, India
mkgourisaria2010@gmail.com

***Abstract*—This paper reports a machine learning model to predict the likelihood of success of android applications. As the android applications are play an important role with in the software industry, it would be a beneficial to study the field. There is currently no archived collection or method for predictingthe success rate of these Android applications. In this research work, we establish a dataset consisting of 30, 000 apps taken from the google play store, third-party apps, and apple storeapps. The dataset is complex and contains about 184 features ofa single application. The data are distributed into two classes malware application and benign applications. The redundant information are dropped from the dataset, following that data cleaning, dimension reduction, and mathematical analysis on the dataset is performed. There are three challenges arise i.e. dataset contain missing value, outliers and class distribution is imbalance. Using the standard techniques the missing value and outliers are treated. For imbalance class distribution various sampling method and cost-sensitive approach are considered. The machine learning algorithms like Logistic Regression (LR), Decision Tree (DT), Support Vector Machine(SVM), and ExtremeGradient Boosting (XG-Boost) are used. It is observed that highest accuracy of 84.44% achieved using ADASYN sampling technique using XG-Boost classifier.**

***Index Terms*—Imbalance data-set, outlier handling, missing value, SMOTE, ADASYN, Android Authenticity Predictions.**

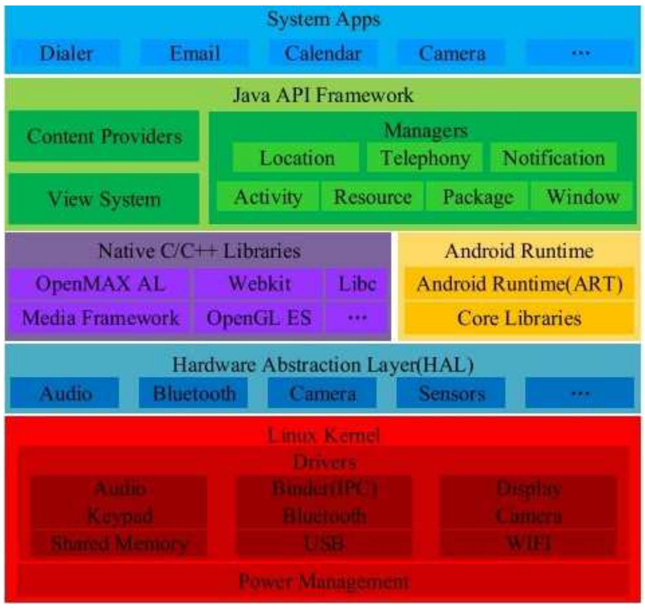


Fig. 1: Android Operating system architecture

I. Introduction

Smart phones make our life easier and more convenient by its capabilities such as accessing the internet, making payments, taking photos, and sharing them, as well as the ability to download various smartphone applications to discover personal or complex features [1], [12]. Smartphones began to be equipped with sensors that allow it to do complicated activities. The number of users of smartphone is increasing day by day. With increased used of this device issue of security also raised. The issue is also resolved with security providing applications. Some the Common types of mobile applications are available in the form of instant messaging apps, downloading apps, anti-virus apps, mobile monitoring apps, and much more [5], [16]. Mostly this phones are use android operating system. The Android operating system is based on the Linux kernel, uses a software stackpile for creating its hierarchical system architecture. As per Fig. 1 [2], Google offers the traditional different layered design of the Android architecture system. These operating systems are supposed to be extremely safe, and they are continually improving and upgrading to address bugs or security flaws. As a result, breaking into systems and doing heinous deeds is extremely tough. Smartphone apps, or apps, on the other hand, pose a danger of compromising privacy [12]. In these context use of machine learning will be time and cost efficient. But very less work were done in this area. Moreover, no efforts have been made to anticipate the non-success or success of mobile applications based on android operating system. In this paper machine learning algorithms are used to prognosticate the non-success or success of mobile applications. The contribution of this paper can be listed as follows:

- Success and failure prediction of Android authentication.
- A dataset is prepared using 30, 000-app store along with 184 features that were considered for dataset implementation.
- The malware applications includes discontinued, suspended, and disconnected apps from the Play Store (Google) collected by the authors
- By calculating the correlation coefficient using heatmap, it appears that all excluded factors are required in prediction and no factors show any dependence on each other and all the factors are of equal value compared to others.
- The generated archive is used for prediction of different Machine Learning strategies for training and modeling. Then, the accuracy score for each Machine Learning technique will be estimated and comparisons of all the strategies with each other will be followed.

II. Literature Review

This section reports some of the benchmark work on malware detection system of Android operating system and different technique to handle imbalance dataset.

A. Background study of Malware detection systems of Android OS

Malware detection systems have evolved over the time.The signature-based technique is the most often used method among them. The signatures of destructive samples are maintained in a database, which is subsequently utilized to recognize malware. To identify unknown malware Machine Learning algorithms have been used. Because of the increase in malware activity in the Android community, extensive research studies have been undertaken toward the identification of malicious Android samples. Researchers are analyzing various aspects to address this issue. Thus numerous static and dynamic improvements in this subject have been done. Static malware analysis entails inspecting the code from a malware vulnerability without running it. Whereas Dynamic malware analysis is used to observe the activity of irregularity (malware) during being performed in a sandboxed environment [8]. A client/server architecture-based strategy for detection of malware on Android devices was given in the research paper [6] which concentratedwhich uses Naive Bayes classification, followed by regression algorithms for improving the classification process. The random forest regression approach achieved excellent results approximately equivalent to 99 percent for malware identification. But due to less work done in data pre-processing and feature selection handling class imbalance is not manoeuvred efficiently and the dataset used were considerably smaller in dimensions. In a similar study [7], the authors created and deployed an Machine Learning model for malware detection purpose on Android smartphones to secure personal and financial data for the ATISCOM program's mobile apps. To lessen the expensive calculations on a mobile device, they transferred data via the device to the server for remote prediction processing. A technique associated with dynamic analysis: T2Droid [15] was created by scientists to detect malware, with a great accuracy rate of roughly 98% and an False Positive Rate of 2% when tested on 160 apps. To launch the system and later the apps, it operates primarily in the secure zone of the ARM TrustZone, which is incorporated in some flagship gadgets as a trusted computing system. As a result, unlike traditional antivirus software, the system is safe and thus it requires complete control over the hardware as well as accessibility to the protected area to implement the program. Incorporating such a mechanism into a lambda device is challenging, if not impossible. MADAM was another method implemented by researchers [14] that is among the most modern and powerful systems for authentic malicious mobile identification using dynamic simulation and machine learning. It categorizes them based on harmful actions discovered at several Android levels: kernel, user, application, and package. It was evaluated on a real device with 2800 malware across 125 distinct families from three datasets, with a detection rate of 96.9 percent. Furthermore, 9804 valid entries were evaluated, revealing an extremely low false positive rate (FPR) of 0.2 percent. In another experiment, [3] researchers created a program application called BAdDroIds, which is accessible on the Google Play Store. They suggested an effective dynamic testing approach based on on-device machine learning. This approach has a 98.9 percent accuracy and a 0.6 percent FPR. BAdDroIds retrieves the authorization stated in the Manifest file and invokes the AAPI i.e. Android Application Program Interface from the DEX code when installing a new application. When the classification process has below 70% certainty, the app allows the user to select whether or not to categorize the installed software as malware. The ML model was developed using the best examination technique from12, 000 samples collected outside of the machine used and prior to testing the 1000 applications loaded on the LG Nexus 5 test smartphone. The process of feature extraction remains the same, however, an Android package analysis takes around 64.474 seconds, and the saved file weight is equivalent to 5539 kilobytes.

Finally, a novel experimental technique developed by scientists known as Android delivers lightweight surveillanceand adaptive analytic tools directly to the device [13]. In the instance of a hacked device, Android's premises are invalid, and it will not be able to detect the virus. Android features a standard antivirus-style blacklist of malware signatures as well as a user whitelist style of malware signatures. For situations that do not fall into either of these categories, behaviors are monitored and recorded in order to construct a behavioral vector that will be contrasted to normal behavior. The findings of 30, 000 apps were tested and it gives us a accuracy of 93% . The vector is generated in 0.53 seconds every 5 seconds. Nevertheless, the functioning of ADroid provides several intriguing concepts that might be enhanced: clustering the vectors can be utilized to decrease its length and

TABLE I: Benchmark work on Android Application analysis

Author and year	Methods or approach	Accuracy	Advantages	Disadvantages
A. Fournier(2021) [8]	Naive Bayes, Regression algorithms	~99%	Good accuracyand results	Used small dataset
S. D. Yalew(2017) [7]	T2Droid	~98%	Operates in ARM TrustZone that is used in flagship models	requires complete control over the hardware
A. Saracino(2018) [15]	MADAM	~97%	One of the Mostpowerful and modern system for malware detection	It is not cost-efficient
S. Aonzo (2020) [14]	BAdDroIds	~97%	Has widespread applicability	Takes more time for operating

for, and incorporating off-device detection as a supplement could enhance speed if the network enables it. Table I shows some benchmark work.

B. Challenges with imbalanced data

The problem of classification of imbalance data is one of the thrust area of machine learning, now-a-days, as maximum of real life data are imbalance like any kind of fraud detection or disease identification etc. The failure to properly take into account their model evaluation metrics is among the most frequent mistakes made by beginners in machine learning when operating with imbalanced data. Simply using a metric like accuracy-score can result in erroneous results becauseit oversimplifies model evaluation, resulting in results thatappear good but are not accurate. The imbalance classificationproblem generally deal using feature selection approach and penalized or cost sensitive approach. Few of the approaches are discussed below.

1) SMOTE: This method [10], [4], is used to avoid over-fitting, that also takes place when exact duplicates of minorities are incorporated to the dataset. A sub - set of information from the minority class is used as an instance, and new synthetically similar objects are created. These fabricated instances are then merged with the original data. The data set is utilized to train the classifier model as a test. When there are outlying inferences in the minority class that emerge in the majority class, SMOTE in not recommended because it creates a line connection with the majority class.

2) ADASYN: ADASYN [11], [9]is a more standard structure that reveals the imperfection of the neighbourhood for each minority inspection by calculating the ratio of majority observational data in the nearest neighbourhood. This impurity ratio is transformed into a probability density function by summing it. As the ratio increases, more synthetic points are produced for that specific point. Hence the number of synthetic observations to be created for Observation is going to bedouble that of observation. So, the drawback of SMOTE such as, border point, noise point, and regular minority points are softer.

III. DATA COLLECTION, DATA PREPOSSESSING AND CLASSIFICATION

In our training set, there are 1772 positive samples and 9076 negative samples, making the positive portion 94%. Hence, the dataset has very high bias and variability. After performing class imbalance, we discard one permission for the first application and resize it to a random portion of between 80% and 100%. As a result, we obtained 1775 samples (malware) and 17289 negative samples (good appli- cations), resulting in 35064 training samples. This method is appropriate in conjunction with mathematical translation to make as many samples as is mentioned, and thus the training class is divided into categories much smaller than the initial training set, therefore the over-sampling method adds many good examples in the machine learning research work. The dataset had some outliers, missing values as well as a lot of subtle details that affected how the findings were assessed. As a result, in order to improve performance and analysis, we have compared various algorithms based on various feature selection methods and methods that are both efficient and cost- effective. We implemented and chose the feature based on SMOTE, ADASYN, Random under Sampling, and Random over Sampling. This section is organized as follows – 1. Dataset Used, 2. Data Exploration, 3.class is balanced, 4. Machine Learning Algorthims used.

A. Dataset Used

The dataset [17] consist of collection of apps from Google play store,Apple store and other 3rd party apps.The dataset contained 184 feature vectors derived from 15,036 applications (5,560 malware apps and 9,476 benign apps). Fig. 2 depicts the distribution of good Applications and Malware Applications from the given dataset.

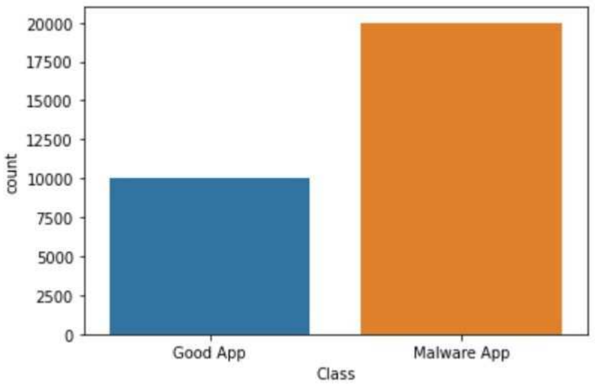


Fig. 2: Class distribution in the dataset

B. Data Exploration

Data analysis and visualization are the most crucial steps in creating a successful machine learning model. It generally includes data analyzing,data cleaning, feature selection, handling missing values, handling class imbalance issue and after which our model will be all set for machine learning algorithms.

In the dataset, there are basically 30,000 apps taken from google play store, apple store and along with other third party apps and this apps are a combination of good and malware apps. The dataset was cleaned, first. the the class distribution was analyzed using counter plot. Analyzing reports, the dataset is skewed towards the malware class. The mean and the median are different from each other thus it is not a normal distribution. Some unwanted columns deleted, like apps de- scription and packages, as they play not role in detecting the authenticity of the application. After deleting the unwanted columns it is highly crucial to remove the duplicates as they can hamper our accuracy and roc auc score. After removing the duplicates and deleting the unwanted columns we comeup to 26,851 rows and 181 columns. Another crucial issue is existence of the missing values. There is only one row have missing values in the apps sector we can simply drop it as by dropping only one row from 30,000 rows probably not affect our performance.

a) Outlier handling:: After treating the missing value, the outlier which is another important phenomenon to dealt. The outliers must be handle in a efficient manner as the outliers decreases our statistical power and increases the variability. The distribution plot is also plotted as it is crucial during assign of the confidence interval. After looking at the box plot and distribution plot we conclude that the box plot has large no of outliers are which will have significant impact on the mean therefore we will use median value of the whole columns to replace the nan values present in dangerous permissioncount. Finally we have all the missing values to the median

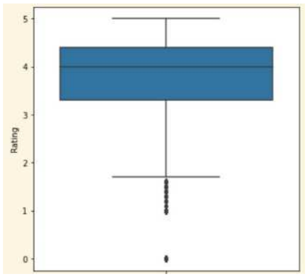
values. Now lets divided our dataset into to categories of categorial and numerical variable and numerical variables are further divided into continuous and discrete variables. The continuous variables are those which have a infinite set of values and discrete variable have infinite set of variables. In our dataset excluding the safe permission count we have all

the variables in the discrete variables as 0 and 1. As there no use of continuous variable as the values present in it are only 0s so its better to remove the unwanted columns. After performing the operation we are left out with 26,851 rows and 158 columns. Now we change the price which is in the float to int along with the dangerous permission count. Now for handling the outliers we have use the Zscore, Percentile, IQR, Isolation Forest but after testing all these it may conculde that

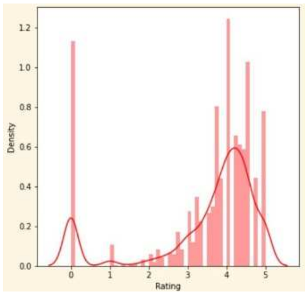
percentile with imputation of median value of the respective column is the best process of removal of outliers. Here not only the outliers are handled but the skewness is also maintained as the best value of skewness and it is fairly symmetrical if its in range -0.5 to 0.5. Fig. 3a and Fig. 3b show the Box plot

and distribution plot of outliers.

b) Dimension reduction: Next major task is dimensionality reduction which basically minimize the no of input variables. It is handle through missing value ratio, low variance filter, high correlation filter, random forest and many more but the random forest suited the best producer of outcome to reduce the dimensionality reduction. Until now we have handled the numerical variable now to handle the categorical variable we use one hot encoding. It basically bring the categorical variable into 0s and 1s form that brings all the categorical variable into a same page and there is no issue of giving priority to a particular column. All the columns are treated equally which leads to getting a perfect predictionof our ml algorithm. In one hot encoding the 0 stands for false/non-existence and 1 stands for true/existence. Now after performing all this above process our data cleaning part is over and we are going to deal with 26, 851 rows and 187 columns. Now the biggest issue that now arises is how to handle class imbalance problem. The issue of class imbalance must be handled otherwise the class will become biased and skewed [18]. This lead to unequal distribution of the training set making theclass biased to a particular thing. Thus handling it become our most priority. This is basically done by various methods like random over and under sampling, SMOTE(Synthetic Minority Oversampling Technique) , ADASYN(Adaptive Synthetic) andtomek link.



(a) Box plot for handling the outliers



(b) Distribution plot for handling the outliers

Fig. 3: Handling Outliers

IV. RESULT AND ANALYSIS

The Performance of the model is mesured based on the parameters as follows *Accuracy*, *F1-Score*, *F2-Score* and ROC Auc score. The equations (1),(2) and (3) demonstratesus the precepts for the above-used metrics in analysing the predictions of different models are as follow:

$$ACCURACY = \frac{TP + TN}{FP + FN + TP + TN} \qquad (1)$$

$$F1Score = \frac{2TP}{2TP + FP + FN} \qquad (2)$$

$$F2Score = \frac{5 * precision * recall}{4 * precision + recall} \qquad (3)$$

Table II gives us the results for these feature selection methods in detail using different parameters for calculation.

And Table III gives the performance of Cost Senstive Based method using different evaluation metrics.

From our final results and implementation we used 4 distinct feature reduction techniques namely Random Under Sampling, Random Over Sampling, ADASYN, SMOTE and cost sensitive based approach and classified the malicious programs using machine learning techniques such as Logistic Regression, Decision Tree, Support Vector Machine, K-Nearest Neighbor, Extreme Gradient Boosting, and Random Forest. The results were then calculated using 5 different parameters:

Feature selection method	Classifi cation algorithm	Accuracy	F1-Score	F2 -Score	Roc-Auc Score	PREDIC TED ROC AUCSC ORE
SMOTE-	SVM	0.7954	0.7732	0.7417	0.7949	NA
	RF	0.8225	0.8013	0.7238	0.8214	0.9097
	DT	0.8361	0.8279	0.8609	0.8380	0.9213
	LOGISTIC	0.7933	0.7771	0.8769	0.7980	0.8701
	XG-BOOST	**0.8864**	**0.8822**	**0.8481**	**0.8866**	**0.9404**
ADASYN	SVM	0.7831	0.7732	0.7394	0.7831	NA
	RF	0.8087	0.7885	0.7126	0.8088	0.8989
	DT	0.8327	0.8246	0.8667	0.8356	0.9184
	LOGISTIC	0.7818	0.7668	0.8238	0.7866	0.8626
	XG-BOOST	**0.8485**	**0.8421**	**0.8798**	**0.8485**	**0.9356**
RANDOM UNDER SAMPL ING	SVM	0.7546	0.7284	0.6466	0.7565	NA
	RF	0.8025	0.7782	0.6807	0.8047	0.8651
	DT	0.8168	0.7997	0.9015	0.8302	0.8775
	LOGISTIC	0.7714	0.7535	0.8348	0.7802	0.8327
	XG-BOOST	**0.8297**	**0.8191**	**0.8917**	**0.8311**	**0.8886**
RANDOM OVER SAMPL ING	SVM	0.7774	0.7503	0.6763	0.7763	NA
	RF	0.8137	0.7868	0.6951	0.8124	0.8804
	DT	0.8303	0.8130	0.8936	0.8397	0.8913
	LOGISTIC	0.7710	0.7474	0.8221	0.7786	0.8426
	XG-BOOST	**0.8481**	**0.8331**	**0.9120**	**0.8472**	**0.9073**

TABLE II: Performance of Feature selection based method

Classifica tion algori thm	accuracy	F1_score	F2_score	Roc_Auc Score	Roc_Auc probability Score
SVM	0.7654	0.8181	0.8338	0.7406	NA
RF	0.8013	0.8489	0.8819	0.7722	0.8951
DT	0.8288	0.8581	0.9023	0.8160	0.8994
LOGISTIC	0.7636	0.8140	0.8106	0.7456	0.8528
XG-BOOST	**0.8314**	**0.8628**	**0.8895**	**0.8291**	**0.9131**

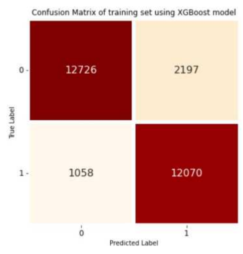
TABLE III: Performance of Cost sensitive based method

Table II shows that the XG-Boost approach performed remarkably well, maintaining a firm and constant connection with the dataset and obtaining an approximate maximum accuracy score of 0.886. It was followed by Decision Tree, which had a maximum accuracy score of 0.836, and Random Forest, which had a score of 0.822. Compared to other models, XG-boost outperformed them in the majority of the measures. The SVM model provided the lowest accuracy with a minimal accuracy score of 0.754, as well as in other parameters in the outcome analysis. Due to the massive size of the dataset, SVM was unable to establish a strong association with the malware detection dataset. Confusion Matrix for XG-Boost and SVM model for their training and testing set are given below in Fig. 4a and Fig. 4b and Fig. 5a and Fig. 5b respectively. False Positive, False Negative, True Positive, andTrue Negative are abbreviated as FP, FN, TP, and TN. The confusion matrix is comprised of these assessment measures. The accuracy of the model is calculated as the number of accurate classifications given by the model divided by theamount of predictions produced. It is one method of evaluatinga model's performance [19], but it is far from the only one. The F1-score is a critical assessment statistic in imbalance dataset. It succinctly summarizes the model's prediction effectiveness bycombining two apparently opposing criteria — accuracy and recall. The F-beta-measure is an extension of the F-measure that includes a beta configuration parameter. A smaller beta number, such as 0.5, provides more emphasis to precisionand less emphasis to recall value in the computation of the score, whereas a greater beta value, such as 2.0, gives less value to precision and more significance to recall. AUC is an abbreviation for "Area Under the ROC Curve". AUC, in other words, measures the complete two-dimensional area beneath the entire ROC curve (think integral calculus) from (0,0) to (1,1). (1,1). AUC is a metric that aggregates performanceacross all categorization criteria. AUC may be interpreted as the likelihood that the model rates a random positive case higher than a random negative example.

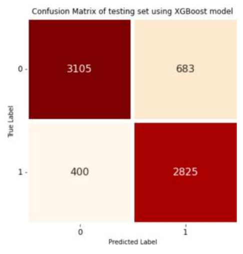
Feature Selection based method produce better result compared to cost-sensitive based method. As result shown in Table II and Table III.

V. CONCLUSION AND FUTURE WORK

This paper reports the success and failure rate of Android applications. Features are retrieved from and merged from a vast collection of high-quality non-computer programmed data and applications. The authorization data is meticulously preprocessed and filtered using the Android Permission Model. Missing values of the datasets are eliminated, which increases

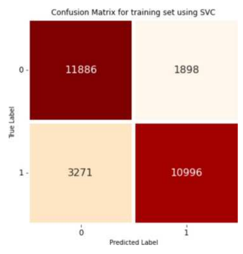


(a) Confusion Matrix for Training XG-Boost Model

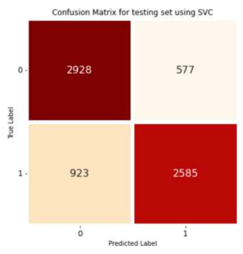


(b) Confusion Matrix dataset for Testing dataset XG-Boost Model

Fig. 4: Confusion Matrix for XG-Boost model



(a) Confusion Matrix for Training dataset SV Classifier



(b) Confusion Matrix for Testing dataset SV Classifier

Fig. 5: Confusion Matrix for SVM model

the effectiveness of our model and makes it more optimal for the machine learning model to use. Re-sampling inaccurate Apps data using the sample technique ADASYN is used to address a problem of class imbalance. The database is processed using decision tree, SVM, K-NN, LR, RF, and XG-Boost. Based on the results, a thorough analysis is conducted, and conclusions are generated that take into account both the precise accuracy and the roc_auc scores. Indicating the level or scope of categorization each of the five models benefited from the final implementation of XG-Boost, which achieved an accuracy of 84.4% and an F-score of 83.0% without over-filling. Additionally, we discovered that the roc auc probability rate was 93.53%, making our model the most accurate available.

References

[1] Worldaˆ€™s most popular mobile operating systems (android vs ios: Market share 2012aˆ C“20‡8). https://ceoworld.biz/2019/01/18/worlds- most-popular-mobile- operating-syste- ms-android-vs-ios-market-share- 2012-2018, 2019.

[2] Platform architecture. accessed. https://developer.android.com/guide/platform , 2020.

[3] Simone Aonzo, Alessio Merlo, Mauro Migliardi, Luca Oneto, and Francesco Palmieri. Low-resource footprint, data-driven malware detection on android. *IEEE Transactions on Sustainable Computing*, 5(2):213–222, 2017.

[4] Jiayao Chen, Hongwei Huang, Anthony G Cohn, Dongming Zhang, and Mingliang Zhou. Machine learning-based classification of rock discontinuity trace: Smote oversampling integrated with gbt ensemble learning. *International Journal of Mining Science and Technology*, 32(2):309–322, 2022.

[5] İbrahim Alper Doğru and Ömer KİRAZ. Web-based android malicious software detection and classification system. *Applied Sciences*, 8(9):1622, 2018.

[6] Arthur Fournier, Franjieh El Khoury, and Samuel Pierre. Classification method for malware detection on android devices. In *Proceedings of the Future Technologies Conference*, pages 810–829. Springer, 2020.

[7] Arthur Fournier, Franjieh El Khoury, and Samuel Pierre. A client/server malware detection model based on machine learning for android devices. *IoT*, 2(3):355–374, 2021.

[8] Ekta Gandotra, Divya Bansal, and Sanjeev Sofat. Malware analysis and classification: A survey. *Journal of Information Security*, 2014, 2014.

[9] Fariha Iffath, Sabrina Jahan Maisha, and Maliha Rashida. Comparative analysis of machine learning techniques in classification cervical cancer using isolation forest with adasyn. In *Proceedings of the International Conference on Big Data, IoT, and Machine Learning*, pages 15–26. Springer, 2022.

[10] Abid Ishaq, Saima Sadiq, Muhammad Umer, Saleem Ullah, Seyedali Mirjalili, Vaibhav Rupapara, and Michele Nappi. Improving the prediction of heart failure patientsaˆ€™ survival using smote and effective data mining techniques. *IEEE access*, 9:39707–39716, 2021.

[11] Taha Muthar Khan, Shengjun Xu, Zullatun Gull Khan, et al. Implementing multilabeling, adasyn, and relieff techniques for classification of breast cancer diagnostic through machine learning: Efficient computer-aided diagnostic system. *Journal of Healthcare Engineering*, 2021, 2021.

[12] Xing Liu, Jiqiang Liu, Sencun Zhu, Wei Wang, and Xiangliang Zhang. Privacy risk analysis and mitigation of analytics libraries in the android ecosystem. *IEEE Transactions on Mobile Computing*, 19(5):1184–1199, 2019.

[13] A Ruiz-Heras, Pedro García-Teodoro, and Leovigildo Sánchez-Casado. Adroid: anomaly-based detection of malicious events in android platforms. *International Journal of Information Security*, 16(4):371–384, 2017.

[14] Andrea Saracino, Daniele Sgandurra, Gianluca Dini, and Fabio Martinelli. Madam: Effective and efficient behavior-based android malware detection and prevention. *IEEE Transactions on Dependable and Secure Computing*, 15(1):83–97, 2016.

[15] Sileshi Demesie Yalew, Gerald Q Maguire, Seif Haridi, and Miguel Correia. T2droid: A trustzone-based dynamic analyser for android applications. In *2017 IEEE Trustcom/BigDataSE/ICESS*, pages 240–247. IEEE, 2017.

[16] Jiyun YANG, Jiang TANG, Ran YAN, and Tao XIANG. Android malware detection method based on permission complement and api calls. *Chinese Journal of Electronics*, 31(4):773–785, 2022.

[17] Suleiman Y. Yerima and Sakir Sezer. Droidfusion: A novel multilevel classifier fusion approach for android malware detection. *IEEE Transactions on Cybernetics*, 49(2):453–466, 2019.

[18] Pawar, K., Jalem, R.S. and Tiwari, V., "Stock market price prediction using LSTM RNN." *In Emerging Trends in Expert Applications and Security*, pp. 493-503. Springer, Singapore, 2019.

[19] Kunal, S., Saha, A., Varma, A. and Tiwari, V., “Textual dissection of live Twitter reviews using naive Bayes”. *Procedia computer science*, 132, pp.307-313, 2018, Elsevier